Google scholar | FAST MUTEX | Search | Advanced Scholar Search

**Scholar** | Articles and patents ▼ | anytime ▼ | include citations ▼ | ✉ Create email alertRe

[PS] Engineering Excellence: DEC OSF/1 Symmetric Multiprocessing
JA Hall - 1994 - users.cs.cf.ac.uk
... Under semaphores (also known as "Mutexes") are **Fast Mutex**, "Recursive **Mutex**" and
"Blocking **Mutex**". Each one of these locks has its place inside the kernel. 3.2.2.1. ... 3.2.2.3. **Fast
Mutex** The **Fast Mutex** allows a single thread to lock the data structure. ...
Cited by 6 - Related articles - View as HTML - All 3 versions

[PDF] A Portable Kernel-Mode Resource Manager on Windows 2000 Platforms
P Hadjidoukas, V Barekas... - Proc. of the Twelth ..., 2000 - Citeseer
... On the other hand, once a **Fast Mutex** has been acquired, the IRQL is raised at
APC_LEVEL, thus the APCs to the thread are blocked. If the thread fails to acquire
the **Fast Mutex**, it is put into a wait state, releasing the processor. ...
Cited by 6 - Related articles - View as HTML - All 3 versions

Towards extremely **fast** context switching in a block-multithreaded processor
W Grunewald... - euromicro, 1996 - computer.org
... The units execute different threads of control. Therefore they access different activation
frames and thus different register sets. A **fast** context switch is realized by simply switching
to another register set. ... Each **mutex** variable exists only once. ...
Cited by 30 - Related articles - All 10 versions

[PDF] Internals of the RT Patch
S Rostedt... - Proceedings of the Linux Symposium, 2007 - kernel.org
... applications. Linux implements a user **mutex** that can create, acquire, and release
the **mutex** lock completely in user space. This type of **mutex** is known as a futex (**fast
mutex**) [1]. The futex only enters the kernel on contention. RT ...
Cited by 36 - Related articles - View as HTML - All 5 versions

**Fast** mutual exclusion for uniprocessors
BN Bershad, DD Redell... - Proceedings of the fifth ..., 1992 - portal.acm.org
Page 1. **Fast** Mutual Exclusion for Uniprocessors Brian N. Bershad ... Mutual ex- clusion
mechanisms such as P, V [Dijkstra 68a] and ac- qutre-**mutex**, re~ease-mut ex [Birrell 91]
are implemented using lower-level operations such as Test-And-Set that ...
Cited by 89 - Related articles - Library Search - BL Direct - All 17 versions

**Fast** algorithms for the simulation of polygonal particles
A Schinner - Granular Matter, 1999 - Springer
Page 1. Granular Matter 2, 35–43 c Springer-Verlag 1999 **Fast** algorithms for the
simulation of polygonal particles Alexander Schinner Abstract Three ... polygons. This
makes **fast** simulations of polygonal assemblies possible. The ...
Cited by 41 - Related articles - BL Direct - All 3 versions

Read/write based **fast**-path transformation for FCFS mutual exclusion
P Jayanti, S Petrovic... - SOFSEM 2005: Theory and Practice of ..., 2005 - Springer
... The winner q of this competition proceeds to compete with the **fast**-path process r through
the 2-**mutex** algorithm. ... When successful, p enters another competi- tion, where it competes
with the **fast**-path process via the 2-**mutex** algorithm (Line 13). ...

Cited by 2 - Related articles - All 4 versions

### GrGen: A **fast** SPO-based graph rewriting tool

R. Geiß, G Batz, D Grund, S Hack... - Graph Transformations, 2006 - Springer
... GrGen: A **Fast** SPO-Based Graph Rewriting Tool 395 AGG VarróDB PROGRES Fujaba
GrGen(PSQL) GrGen(SP) size time 1ms 10ms 100ms 1s 1min 1h 100 250 500 750 1000 Fig.
4. Runtime of STS **mutex** benchmark (multiplicity optimizations off, parameter passing off **...**
Cited by 82 - Related articles - BL Direct - All 9 versions

### [PDF] Multi Thread programming

A Negri, DA Scannicchio, F Touchard... - 2001 - atlserver.pv.infn.it
... state. pthread_mutex_trylock behaves identically to pthread_mutex_lock, except that
it does not block the calling thread if the **mutex** is already locked by another thread
(or by the calling thread in the case of a **fast mutex**). Instead **...**
Cited by 1 - Related articles - View as HTML - All 4 versions

### Software generation of practically strong random numbers

P Gutmann - Proceedings of the Seventh USENIX Security ..., 1998 - usenix.org
... To do this we use two methods, a **fast** randomness poll which executes very quickly and gathers
as much random (or apparently random) information as quickly as possible, and a slow poll which
can take a lot longer than the **fast** poll but which performs a more in-depth search **...**
Cited by 47 - Related articles - All 35 versions

⊠ Create email alert

Goooooooooog l e ▶

Result Page:    1 2 3 4 5 6 7 8 9 10    **Next**

FAST MUTEX                          Search

Go to Google Home - About Google - About Google Scholar

©2011 Google